ANGULAR

1 . Where angular is used…??

YouTube TVs is the most popular example that uses Angular.

2 . Package-lock.json…??

**In this is package all those file or packages or dependency will be displayed that is installed in your system.**

3 . Package.json…??

**In this is package all those file or packages or dependency will be displayed that is important or required for your application.**

4 . What are decorators…??

Decorators are the features of typescript and are implemented as function. The name of the decorator starts with @ symbol.

Decorators are simply functions that return functions. These functions supply metadata to Angular about a particular class, property, value, or method…

Decorator are invoked at runtime.

Decorators allows as to execute functions. For example, @components executes the component function imported from Angular 7.

Ex :

@NgModule to define module.

@Component() to define components.

@Injectable to define service.

@input and @output to define the property, that send and receive data from DOM.

* **Class decorators like @Component, @NgModule**
* **Property decorators like @Input and @Output**
* **Method decorators like @HostListener**
* **Parameter decorators like @Injectable**

5 . What are Directives…??

**Directives are the elements which change the appearance or behavior of the DOM elements.**

**There are three types of Directives manly:**

**1. Components Directives :**

**Without component directive we cannot imagine angular.**

**2. Structural Directives :**

**i. \*ngIf.**

**ii. \*ngFor**

**iii. \*ngSwitch**

**3. Attributes Directive :**

**i. NgStyle**

**ii. NgClass**

**6. Custom Pipes……???**

import { [Pipe](https://angular.io/api/core/Pipe), [PipeTransform](https://angular.io/api/core/PipeTransform) } from '@angular/core';

@[Pipe](https://angular.io/api/core/Pipe)({name: 'exponentialStrength'})

export class ExponentialStrengthPipe implements [PipeTransform](https://angular.io/api/core/PipeTransform) {

transform(value: number, exponent = 1): number

{

return Math.pow(value, exponent);

}

}

1. **Why we need node.js in Angular Application……???**

Angular does not need Node.js directly and it is not mandatory to use Node.js. But you will need Node.js for all the build and development tools.

For an example these are few reasons that you need Node.js for building an Angular app,

* npm (node package manager) comes with Node.js by default and it allows you to manage your dependencies. So, you don’t have to worry for operations like adding a dependency, removing some, updating your package.json.
* npm gives you angular cli or ng cli (angular command-line interface) which is a great tool for building your application easily
* Node.js allows you to spin up a lightweight web server to host your application locally in your system.

**8. Differences between Annotation and Decorator….???**

|  |  |
| --- | --- |
| **Annotation** | **Decorator** |
| **Used by Traceur compiler** | **Used by Typescript compiler** |
| **Annotations are only metadata set on the class using the Reflect Metadata library.** | **Decorator corresponds to a function that is called on the class.** |
| **Annotations are used for creating an attribute annotations that stores array.** | **Decorator is a function that gets the object that needs to be decorated.**  **A decorator is a function that adds metadata to a class, its members, or its method arguments. A decorator is just a function that gives you access to the target that needs to be decorated.** |
| **They are Hard-coded** | **They are not Hard-coded** |
| **Exp. Imports for Annotations: import {ComponentAnnotation as Component} from ‘@angular/core’;** | **Exp. Imports for Decorators: import {Component} from ‘@angular/core’;** |

Difference between Let Vs Var :

|  | **var** | **let** |
| --- | --- | --- |
| **1.** | The var is a keyword that is used to declare a variable | The let is also a keyword that is used to declare a variable. |
| **2.** | Syntax -:  **var name = value;** | Syntax -:  **let name = value;** |
| **3.** | The variables that are defined with **var**statement have function scope. | The variables that are defined with **let**statement have block scope. |
| **4.** | We can declare a variable again even if it has been defined previously in the same scope. | We cannot declare a variable more than once if we defined that previously in the same scope. |
| **5.** | Hoisting is allowed with **var**. | Hoisting is not allowed with **let**. |
| **6.** | Example -:  **var websitename = “geeksforgeeks”;** | Example -:  **let x = 69;** |
| **7.** | var is an ECMAScript1 feature. | let is a feature of ES6. |
| **8.** | Its supported browsers are: Chrome, Internet Explorer, Microsoft Edge, Firefox, safari, opera | Its supported browsers are -: Chrome49, Microsoft Edge12, firefox44 , safari11, opera36 |